سوال ۱) سیستمی را طراحی و پیاده سازی نمایید که مشخصات پذیرنده متناهی قطعی (DFA) را از فایل ورودی دریافت و بررسی می کند که رشته های وارد شده توسط کاربر در DFA پذیرفته میشود یا نمیشود.

جواب) این کد شامل ۵ کلاس است :

۱. Main
۲. SelectFile
۳. Splitter
۴. State
۵. ReadFile

- **کلاس SelectedFile**

در این کلاس با استفاده از JFileChooser فایل txt را کاربر انتخاب میکند و برنامه چک میکند تا حتما فایل txt باشد ، در صورت نبودن پیغام خطا میدهد و از کاربر میخواهد فایل درست را انتخاب کند. بعد از دریافت فایل درست ، آماده ی خواندن آن میشویم.

- **کلاس Splitter**

بعد از خواندن هر خط به دلیل وجود چندین متغیر که باید از هم جدا شوند، این کلاس ساخته شده است. که با استفاده از دستور split در هر خط متغیر های مورد نظر را که با فاصله از هم جدا شده اند را در ارایه ای می ریزد.

- **کلاس State**

در این کلاس برای انتقال ۴ متغیر اصلی داریم:

currentState (حالت اولیه)

hashmap از alphabet به عنوان کلید اصلی و آرایه ای از transition ها در واقع حالت بعدی (حرف الفبا و حالات نهایی)

Boolean های start و finish که حالت های شروع کننده و نهایی را به ما نشان میدهند.

که اگر ماشین قطعی نباشد میتواند از یک حالت با یک حرف الفبا به بیش از یک حالت برود یا ممکن است با توجه به حروف الفبایی که پیش تر کاربر تعریف کرده است ، از currentState با این الفبا انتقالی تعریف نشده باشد.

به همین دلیل حالت دوم به صورت hashmap تعریف شده است.

- **کلاس ReadFile**

در اینجا در ابتدا که از کاربر میخواد فایل را انتخاب کند. سپس شروع به خواندن فایل میکند.

ابتدا خط اول که حروف را به ما میدهد و در ارایه ای به نام alphabets ذخیره میشوند.

سپس تمام حالت های ممکنه را در یک hashmap به اسم states ذخیره میکنیم. اسم هر حالتی به عنوان کلید اصلی و value ویژگی های آن حالت و انتقالات آن را ذخیره میکند.

بعد از خواندن خط سوم و چهارم قسمت value، states را به روزرسانی میکنیم به این صورت که اگر حالت شروع کننده باشد Boolean start را true میکنیم و حالت های نهایی را Boolean finish را true میکنیم.

سپس میخواهیم انتقال ها را بخوانیم.

تا زمانی که خط بعدی وجود داشته باشد و به null نرسیده باشیم این فرایند ادامه دارد.

بعد از خواندن خط و جدا کردن اجزای آن ابتدا چک میکنیم که این حالت در حالت های ما موجود باشد اگر بود چک میکنیم که انتقالی از قبل وجود داشته است یا نه اگر داشته باشید flag را true میکنیم به این معنی که در ماشین DFA نمیتواند از یک حالت با یه حرف الفبا یه چندین حالت دیگر منتقل شود.

بعد از خواندن انتقال ها آنها را به hashmap states اضافه میکنیم و قبل از آن حتما چک میکنیم که این حرف الفبا که برای انتقال داده شده قبلا حتما تعریف شده باشد.

- **کلاس Main**

در کلاس main ابتدا فایل را دستور میدهیم بخواند. سپس اگر flag ، false بود چک میکنیم که از همه ی حالت ها با همه ی حروف تعریف شده حتما حالت بعدی وجود داشته باشد در غیر اینصورت flag ، true میشود.

و در آخر اگر flag ، true بود میگوییم که این رشته توسط ماشین DFA قابل قبول نیست در غیر اینصورت پذیرفته میشود.

سوال ۲) سیستمی را طراحی و پیاده سازی نمایید که پذیرنده متناهی غیر قطعی (NFA) را به پذیرنده متناهی قطعی (DFA) تبدیل نماید.

جواب) این کد شامل ۸ کلاس است :

۱. Main
۲. SelectFile
۳. Splitter
۴. State
۵. ReadFile
۶. LambdaClosure
۷. Conversion
۸. PrintStates

- **کلاس SelectedFile**

در این کلاس با استفاده از JFileChooser فایل txt را کاربر انتخاب میکند و برنامه چک میکند تا حتما فایل txt باشد ، در صورت نبودن پیغام خطا میدهد و از کاربر میخواهد فایل درست را انتخاب کند. بعد از دریافت فایل درست ، آماده ی خواندن آن میشویم.

- **کلاس Splitter**

بعد از خواندن هر خط به دلیل وجود چندین متغیر که باید از هم جدا شوند، این کلاس ساخته شده است. که با استفاده از دستور split در هر خط متغیر های مورد نظر را که با فاصله از هم جدا شده اند را در ارایه ای می ریزد.

- **کلاس State**

در این کلاس برای انتقال ۵ متغیر اصلی داریم:

currentState (حالت اولیه)

hashmap از alphabet به عنوان کلید اصلی و آرایه ای از transition ها در واقع حالت بعدی (حرف الفبا و حالات نهایی)

Boolean های start و finish که حالت های شروع کننده و نهایی را به ما نشان میدهند.

و یک stateName که در صورتی که اسم اسم از چند متغیر تشکیل شده بود آرایه ای string که در واقع همان نام state هامون است را در اینجا ذخیره میکنیم.

که اگر ماشین قطعی نباشد میتواند از یک حالت با یک حرف الفبا به بیش از یک حالت برود یا ممکن است با توجه به حروف الفبایی که پیش تر کاربر تعریف کرده است ، از currentState با این الفبا انتقالی تعریف نشده باشد.

به همین دلیل حالت دوم به صورت hashmap تعریف شده است.

- **کلاس ReadFile**

در اینجا در ابتدا که از کاربر میخواد فایل را انتخاب کند. سپس شروع به خواندن فایل میکند.

ابتدا خط اول که حروف را به ما میدهد و در ارایه ای به نام alphabets ذخیره میشوند. و در اخر حرکت لاندا را خودمان به آرایه اضافه میکنیم.

سپس تمام حالت های ممکنه را در یک hashmap به اسم states ذخیره میکنیم. اسم هر حالتی به عنوان کلید اصلی و value ویژگی های آن حالت و انتقالات آن را ذخیره میکند.

بعد از خواندن خط سوم و چهارم قسمت value، states را به روزرسانی میکنیم به این صورت که اگر حالت شروع کننده باشد Boolean start را true میکنیم و حالت های نهایی را Boolean finish را true میکنیم.

سپس میخواهیم انتقال ها را بخوانیم.

تا زمانی که خط بعدی وجود داشته باشد و به null نرسیده باشیم این فرایند ادامه دارد.

بعد از خواندن خط و جدا کردن اجزای آن ابتدا چک میکنیم که این حالت در حالت های ما موجود باشد اگر بود چک میکنیم که انتقالی از قبل وجود داشته است یا نه بعد از خواندن انتقال ها آنها را به hashmap states اضافه میکنیم و قبل از آن حتما چک میکنیم که این حرف الفبا که برای انتقال داده شده قبلا حتما تعریف شده باشد.

- **کلاس Main**

در کلاس main ابتدا فایل را دستور میدهیم بخواند.

سپس تبدیل های لازم روی آن انجام می پذیرد. (کلاس Conversion را صدا میزنیم.)

و در آخر حالت ها را چاپ میکنیم.(ماشین DFA تولید شده را چاپ میکنیم.)

- **کلاس PrintStates**

هر حالت را با انتقال ها و حالت بعدی اش چاپ میکند.

- **کلاس Conversion**

ماشین NFA داده شده به آن را به DFA تبدیل میکند.

ابتدا NFA باحرکت لاندا را به NFA تبدیل میکند سپس آن را به DFA تبدیل میکند.

برای اینکار در ابتدا کلاس LambdaClosure را صدا میزند.

سپس برای هر حالت انتقال های آن را چک میکند اگر انتقال هایی که به آن میرود از طریق حرف مورد نظر به عنوان حالت وجود نداشت آن حالت را به حالت های قبلی اضافه میکند و انتقال های مورد نظرش را به آن اضافه میکند. مثلا اگر حالت جدید ما q۰,q۱ باشد انتقال هایی که در q۰ و q۱ هستن را به این حالت اضافه میکند.این کار را انقدر ادامه میدهیم تا حالت جدیدی برای اضافه کردن وجود نداشته باشد.(در واقع دلتا استار را حساب میکنیم.)

یک حالت دیگر هم در نظر میگیریم به اسم trap در صورتی که حالتی وجود داشته باشد که با یکی از حرف های تعریف شده ، انتقالی برایش وجود نداشته باشد به این حالت منتقل میشود.

و تمام انتقال های trap به روی خودش است.

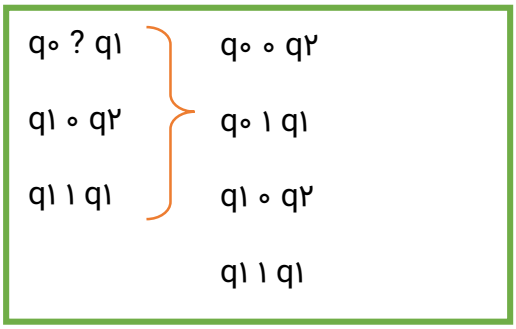
هنگام اضافه کردن حالت های جدید مطمئن میشویم حالت های قبلی را دوباره اضافه نکند مانند q۰,q۱ و q۱,q۰ .(با استفاده از sort کردن رشته ها)

- **کلاس LambdaClosure**

این کلاس برای تبدیل NFA باحرکت لاندا را به NFA است.

به این صورت که حالت هایی که دارای انتقال با حرکت لاندا هستند را انتخاب میکند و حالت دوم آنها را نیز انتخاب میکند ، انتقال هایی که برای حالت دوم وجود دارد را به این حالت اضافه میکند.

مثلا



تبدیل میکند.و در آخر انتقال با حرکت لاندا را از انتقال هایش حذف میکند.

بعد از چک کردن تمام حالت ها حرکت لاندا را از آرایه ی حروفمون حذف میکنیم.